Analysis of Linear Trends in USA Aiport Temperatures

Andy Pickering

10/12/2016

I love looking at weather data and have experimented in the past with some of the airport weather station data available from http://www.wunderground.com. I saw this little analysis on R-bloggers https://www.r-bloggers.com/annual-mean-temperature-trends-12-airports/ and thought it would be interesting to look at trends from stations across the US and visualize the results on a map.

```
# Setup, load packages
rm(list=ls())
library(weatherData)
suppressPackageStartupMessages(library(lubridate))
library(ggplot2)
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(maps))
library(RColorBrewer)
library(magrittr)
library(leaflet)
```

First we need a list of the weather stations we will use. Luckily the weatherData package already contains a compiled data frame 'USAirportWeatherStations' with all the stations and info about them. The 'airportCode' field is what will identify the station when we request the data through the wunderground API.

```
head(USAirportWeatherStations)
```

```
##            Station State airportCode   Lat     Lon Elevation   WMO
## 1          Central    AK        PARL 65.57 -144.80       292 99999
## 2             Atka    AK        PAAK 52.22 -174.20        17 99999
## 3         Buckland    AK        PABL 65.99 -161.12         0 99999
## 4 Portage Glacier    AK        PATO 60.79 -148.83        29 99999
## 5         Nivalina    AK        PAVL 67.73 -164.55         3 70148
## 6          Golovin    AK        PAGL 64.55 -163.05         8 70199
```

I want to fit a long-term trend, so i'll get data starting in 1980 (a lot of the stations seem to go back this far).

```
# Years we will get data for
year_list=1980:2016
st_list <-USAirportWeatherStations$airportCode
# station KMMO 1993 doesn't work, skip
st_list <- st_list[-which(st_list=="KMMO")]
```

Let's first plot all the station locations. For this analysis, I'll just use those in the continental/lower 48 US states. Including others like Alaska, Hawaii, Virgin Islands etc. makes the map very large and difficult to see. The leaflet pacakge makes a nice interactive map (try zooming and clicking on locations!).

```
# just plot all station locations
cont_us <- USAirportWeatherStations %>%
        filter(State!="AK" & State!="MP" & State!="PR" & State!="HI" & State!="VI" & State!="GU")

m <- leaflet() %>%
        setMaxBounds(-125,23,-67,50)%>%
        addTiles() %>%  # Add default OpenStreetMap map tiles
        addCircleMarkers(lng=cont_us$Lon,lat=cont_us$Lat,popup=cont_us$Station,radius=2,fill=TRUE,color=
m  # Print the map
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed, please
```

The next bit of code downloads the data from http://www.wunderground.com. This is the most time-consuming part of the analysis. It appears that the wunderground API only lets you download a year of data at a time. With my laptop and wifi speed, I estimate it takes about 2-3 sec to download each 1 year data file. For 36 years of data (1980-2016) at about 1600 stations, that works out to about 48 hours! Luckily it is easy to write a loop to do this, and I just let it run overnight or in the background until it was done.

```
# Define function to download daily weather for 1 year.
# I had trouble getting the functions in the weatherData package to work, but once we have the formula

get_yearly_weather <- function (year,st_code){
        url <- paste0("http://www.wunderground.com/history/airport/",st_code,"/",year,"/1/1/CustomHisto
}

# Define a function to download the weather data for a given station code for each year
get_all_years <- function(st_code){
        for (i in seq_along(year_list)) {
                year <- year_list[i]
                savefile<-file.path("~/AirportTemps/Data",paste0("wea",st_code,year,".csv"))
                if (file.exists(savefile)){
                        print(paste(savefile," already downloaded"))
                }else{
                        url=get_yearly_weather(year,st_code)
                        print(paste("Getting weather data for ",st_code," for year ",year))
                        download.file(url,savefile)
                }
        }
}

# Apply that function to station list to get data for all stations
tryapply(st_list, get_all_years)
```

Ok, we finally have all the yearly data files downloaded. Now i'll combine the yearly files (1980-2015) for each station into a single combined file. I also do some cleaning and remove bad temperature values.

```
# Remove bad temperature values (marked as -99999 in data)
rm_bad_wea_values <- function(dat){
        dat$Min.TemperatureF[which(dat$Min.TemperatureF==-99999)]<-NA
        dat$Max.TemperatureF[which(dat$Max.TemperatureF==-99999)]<-NA
        dat$Mean.TemperatureF[which(dat$Mean.TemperatureF==-99999)]<-NA
        dat<-subset(dat,Mean.TemperatureF>-50);
        dat<-subset(dat,Mean.TemperatureF<150);
        dat
}

# Define a function to load and combine all years for a station into one dataframe
combine_years_wea <- function(st_code){
        savefile <-file.path( "~/AirportTemps/Data",paste0("wea",st_code,"combined.csv"))
        print(paste("combining years for station ", st_code))
        # Load csv files for each year and combine into a single data frame
        year_list=1980:2015
        dat_all=data.frame()
        for (i in seq_along(year_list)) {
                year <- year_list[i]
                #print(year)
                dat <- read.csv( file.path( "~/AirportTemps/Data",paste0("wea",st_code,year,".csv")))
                dat<-rm_bad_wea_values(dat)
                dat$dd <- ymd(dat[,1])
                dat_all=rbind(dat_all,dat)
        }
        # Check to make sure we have data (some files are empty)
        if (nrow(dat_all)>(20*365)){
                # Save csv with combined data...
                write.csv(dat_all,savefile)
        }else{
                print(paste("station ",st_code," doesn't have data"))
        }
}

# some stations don't work, so need tryapply()
tryapply(st_list, combine_years_wea)
```

Ok, we now have a single file for each station with all the data. I loop through and fit a linear trend vs time for each one, and store all the results in a dataframe. I'll save the trend (coefficent) from the fit, and the p-values to evaluate how reliable the fit was.

```
library(broom)
# save data frame with fit coeffs, p value etc.
temp_fits<-data.frame()
for (i in seq_along(st_list)) {
        fname<-file.path( "~/AirportTemps/Data",paste0("wea",st_list[i],"combined.csv"))
        if (file.exists(fname)){
                print(paste("Fitting to ",st_list[i]))
                dat<- read.csv(fname)
                dat$dd <- ymd(dat$dd)
                fit1 <- lm(Mean.TemperatureF~dd,data=dat)
                fit_tidy <- tidy(fit1)
#                summary(fit1)
                temp_fits<-rbind(temp_fits,data_frame(airportCode=st_list[i],trend=fit_tidy$estimate[2]
        }else{
                print(paste("combined file for station ", st_list[i], " doesn't exist, skipping"))
        }
}

# save a csv file with the fit results
write.csv(temp_fits,file.path( "~/AirportTemps/Data","temp_fits.csv"))
```

Let's look at the results. First i'll define a little function to simplify loading the data.

```
# function to load combined 1980-2016 csv file for specified station
load_combined<-function(st_code){
        fname<-file.path( "~/AirportTemps/Data",paste0("wea",st_code,"combined.csv"))
        dat<- read.csv(fname)
        dat$dd <- ymd(dat$dd)
        dat
}
```

```
temp_fits <- read.csv(file.path( "~/AirportTemps/Data","temp_fits.csv"))

# convert trend to deg/decade
temp_fits$trend <- temp_fits$trend*365*10
```

How many fits have significant/non sig. p-values?

```
ig<-which(temp_fits$pval<0.05)
ib<-which(temp_fits$pval>0.05)
print(paste(length(ig)," fits have p-values that are significant(<0.05)"))
```

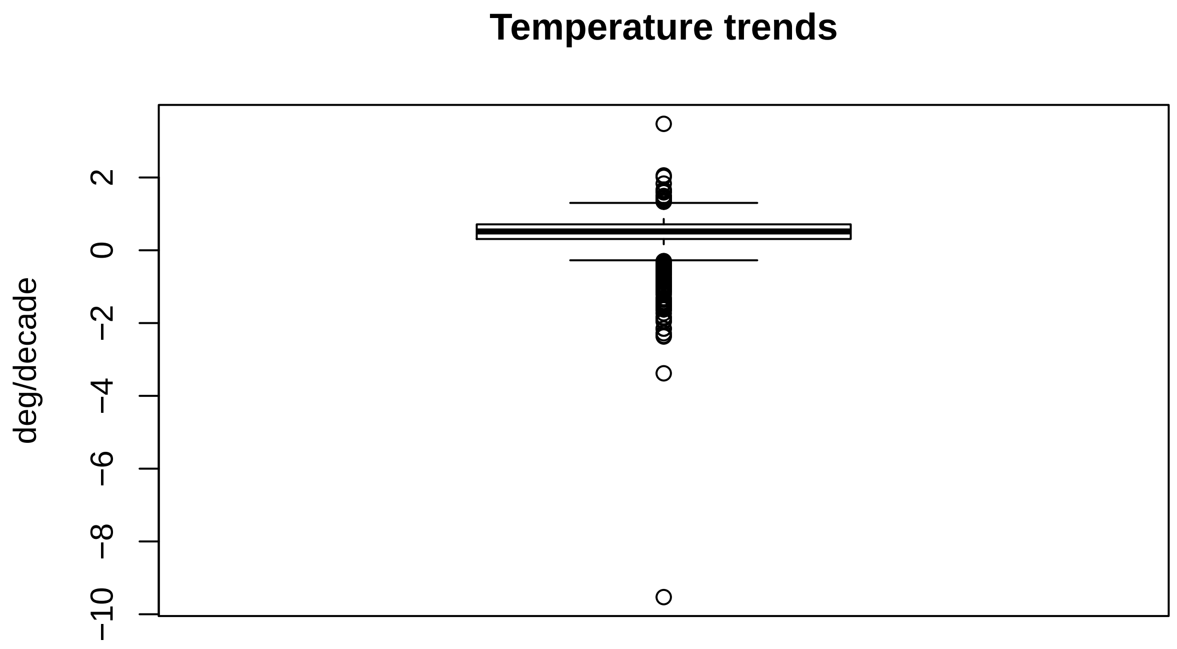
```
## [1] "674  fits have p-values that are significant(<0.05)"
```

```
print(paste(length(ib)," fits have p-values that are not significant(>0.05)"))
```

```
## [1] "350  fits have p-values that are not significant(>0.05)"
```

Look at the distribution of trends.

```
boxplot(temp_fits$trend[ig],main="Temperature trends",ylab= "deg/decade")
```

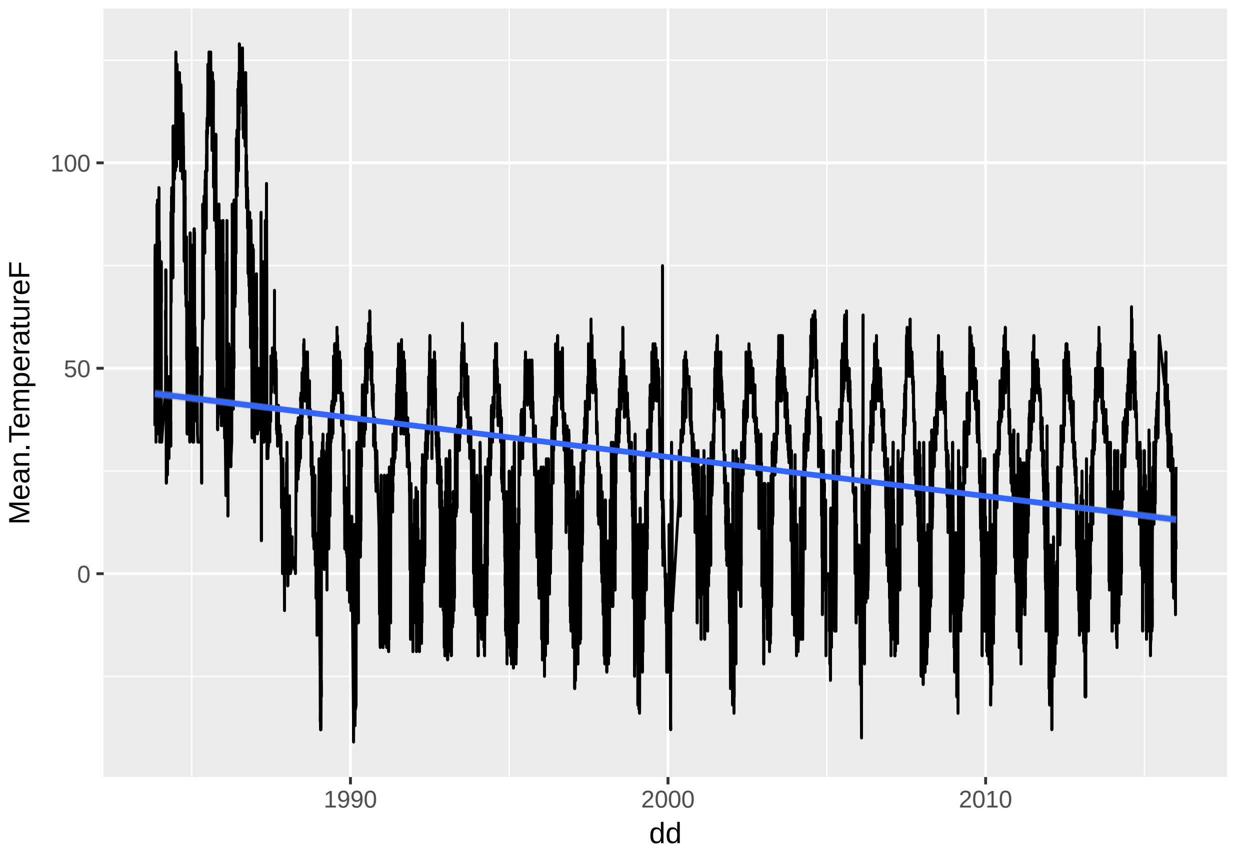


Most of the trends seem reasonable (magnitudes of about 2 degrees per decade), but there a few outliers. Lets look at those and see what's going on.

```
iout <- which(abs(temp_fits$trend)>5)
temp_fits[iout,]
```

```
##   X airportCode     trend          pval
## 5 5        PASH -9.528885 6.728656e-252
```

```
datout <- load_combined(temp_fits$airportCode[iout])
g<-ggplot(datout,aes(x=dd,y=Mean.TemperatureF))+geom_line()+geom_smooth(method="lm")
g
```

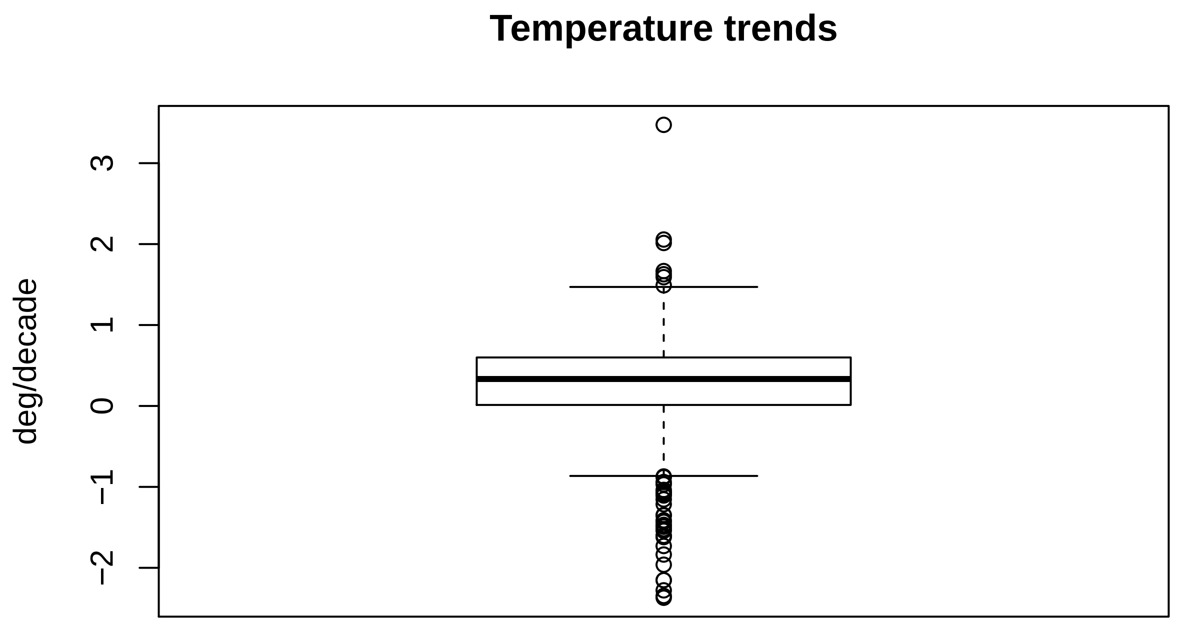


It's obvious from the plot that something is wrong with the data for this station; there is a huge offset after the first ~5 years. Maybe the station was moved, or mislabeled? We'll remove this station (note it is in Alaska, so I wouldn't have plotted it in the end anyways).

```
temp_fits <- temp_fits[-iout,]
```

Look at the distribution of trends again with the big outlier removed:

```
boxplot(temp_fits$trend[ig],main="Temperature trends",ylab= "deg/decade")
```

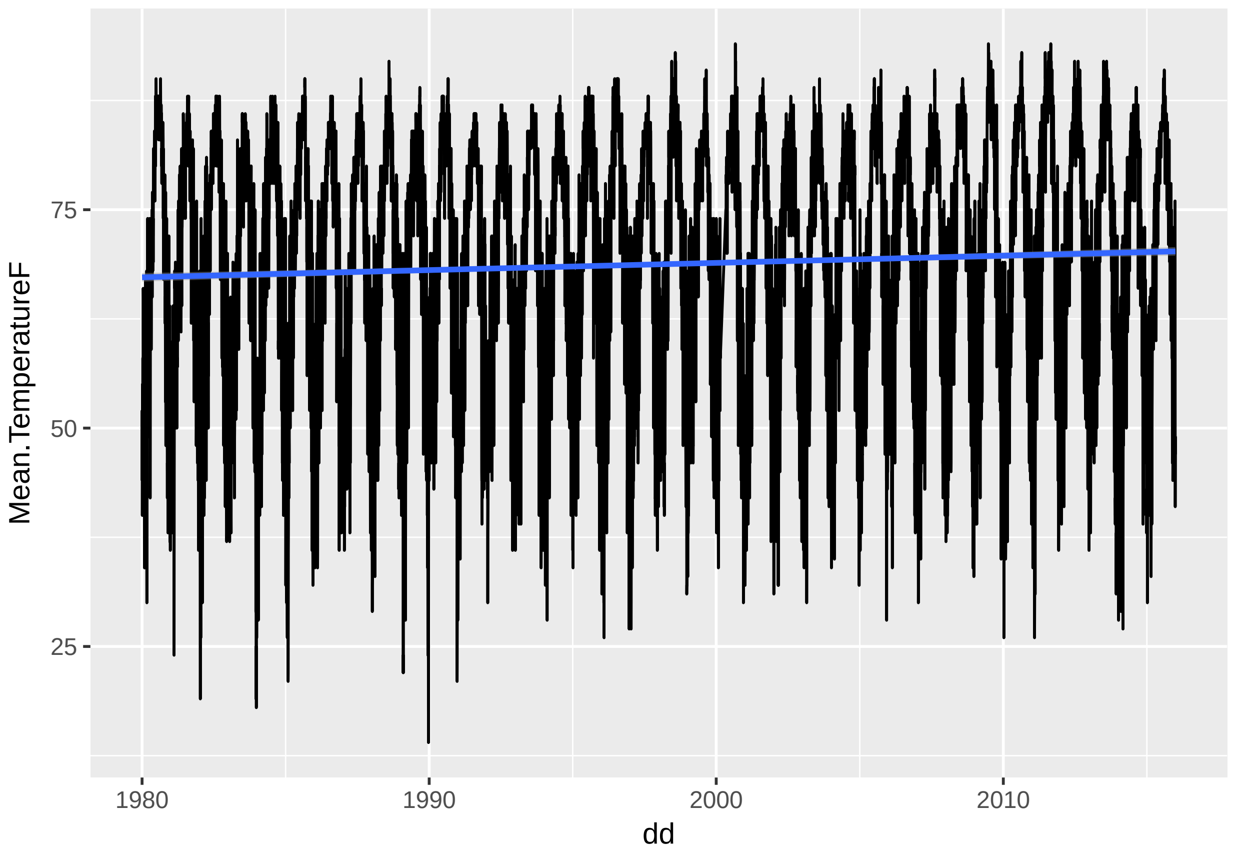


Let's look at a good fit also, to convince ourselves we are doing this right:

```
id <- ig[550]
temp_fits[id,]
```

```
##       X airportCode     trend         pval
## 851 851        KCLL 0.8246019 7.447468e-12
```

```
datgood <- load_combined(temp_fits$airportCode[id])
g<-ggplot(datgood,aes(x=dd,y=Mean.TemperatureF))+geom_line()+geom_smooth(method="lm")
g
```



Ok this looks better. Note I haven't removed any of the obvious seasonal cycle. My reasoning is that i'm interested in the long-term trend, and if there is a constant seasonal cycle it should average itself out (ie if the only variability in temperature was a seasonal cycle, the fit should give a trend of zero). In the future I plan to try fits over different time periods, and also with the seasonal cycle removed, to see how that affects the results.

Ok, now it's time to make the awesome map i've been envisioning. First i'll join the fit results with the station info so we have the locations and data together.

```
# Only keep stations with significant p-values
temp_fits_good <- temp_fits %>%
        filter(pval<0.05)
# Join to data frame with station info
results2 <- inner_join(temp_fits_good,USAirportWeatherStations,by="airportCode")
```

```
## Warning in inner_join_impl(x, y, by$x, by$y, suffix$x, suffix$y): joining
## factor and character vector, coercing into character vector
```

```
# Keep only continental and lower 48 states (map doesn't look good with others)
results2<- results2 %>%
        filter(State!="AK" & State!="MP" & State!="PR" & State!="HI" ) %>%
        arrange(desc(trend))

# Take a look at the largest, smallest trends
#sortdat<-arrange(results2,desc(trend))
head(results2)
```

```
##     X airportCode    trend         pval      Station State   Lat     Lon
## 1 575        KOLF 3.473326 5.393107e-34   Wolf Point    MT 48.10 -105.58
## 2 704        KHTO 2.013744 4.799912e-44 East Hampton    NY 40.97  -72.25
## 3 514        KMOX 1.831821 8.707694e-07       Morris    MN 45.57  -95.97
## 4 689        KRNO 1.666679 7.237582e-38         Reno    NV 39.50 -119.78
## 5 366        KHUT 1.636124 1.180091e-21   Hutchinson    KS 38.07  -97.87
## 6 522        KRGK 1.627320 5.429190e-06     Red Wing    MN 44.58  -92.48
##   Elevation   WMO
## 1       605 99999
## 2        55 99999
## 3       344 99999
## 4      1341 72488
## 5       470 99999
## 6       239 99999
```

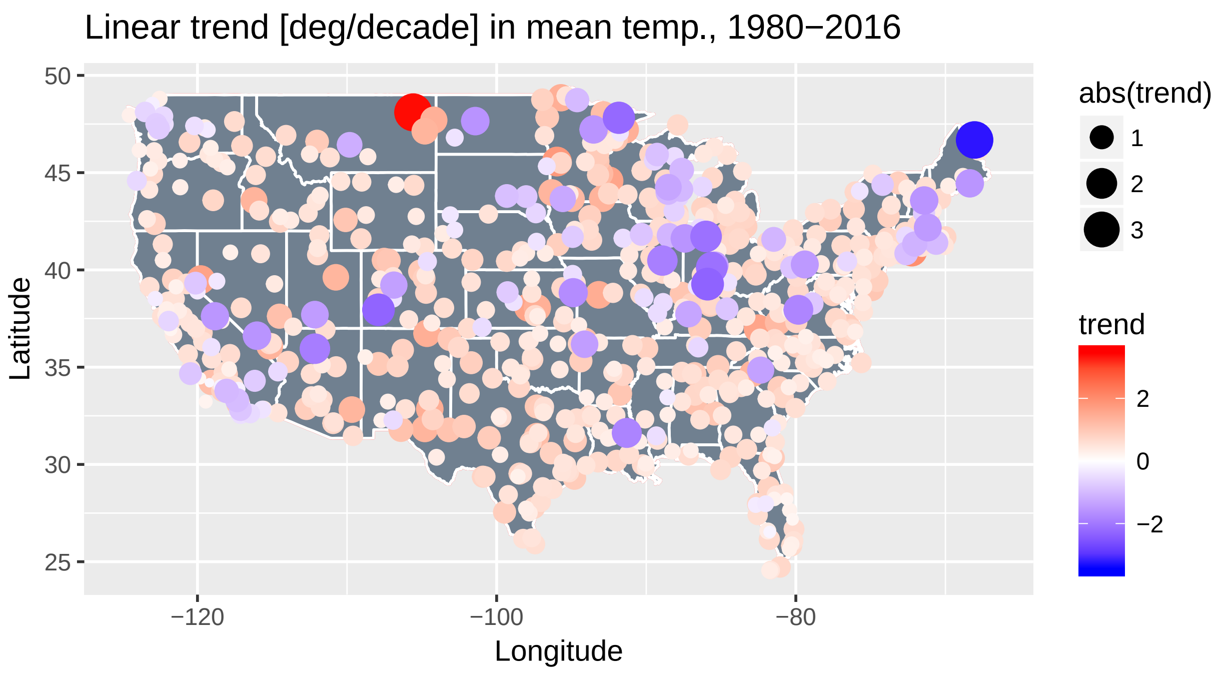
```
tail(results2)
```

```
##       X airportCode     trend         pval      Station State   Lat
## 650 343        KEKM -2.152481 5.944056e-38      Elkhart    IN 41.72
## 651 340        KAID -2.152536 2.705784e-13     Anderson    IN 40.12
## 652 498        KELO -2.278304 7.698791e-21          Ely    MN 47.82
## 653 182        KTEX -2.347279 4.200821e-34    Telluride    CO 37.95
## 654 341        KBAK -2.369682 1.266927e-24     Columbus    IN 39.27
## 655 434        KPQI -3.380191 5.686474e-66 Presque Isle    ME 46.68
##         Lon Elevation   WMO
## 650  -86.00       237 99999
## 651  -85.62       280 99999
## 652  -91.83       443 99999
## 653 -107.90      2767 99999
## 654  -85.90       200 99999
## 655  -68.05       146 72713
```

Now I can plot the map, with the size and color of circles proportional to the trends. Red colors indicate positive (warming) trends, and blue colors indicate negative (cooling) trends.

```
## Plot map with circles proportional to trend
usa <- map_data("usa")
states <- map_data("state")

ggplot() +
        geom_polygon(data = usa, aes(x=long, y = lat, group = group), fill=NA, color="red") + coord_fixe
        geom_polygon(data=states,aes(x = long, y = lat, group = group), fill="slategray",color = "white
        guides(fill=FALSE) +  # leave off the color legend
        geom_point(data = results2, aes(x = Lon, y = Lat,size = abs(trend),color=trend)) +
        scale_color_gradient2(midpoint=0, low="blue", mid="white",high="red", space ="Lab" ,limits=c(-3
        ggtitle("Linear trend [deg/decade] in mean temp., 1980-2016") +
        labs(x="Longitude",y="Latitude")
```



```
# Make the same map w/ leaflet
library(leaflet)

# make color palette. I want to use the diverging red-blue palette, but need to reverse it so blue is n
pal_blrd <- brewer.pal(10,"RdBu")
pal <- colorNumeric(palette=rev(pal_blrd),domain=c(-3.5,3.5))

# Also make an interactive leaflet map
m <- leaflet(data=results2) %>%
        setMaxBounds(-125,23,-67,50)%>%
        addTiles() %>%  # Add default OpenStreetMap map tiles
        addCircleMarkers(~Lon,~Lat,popup=paste(results2$Station,round(results2$trend,digits=2)),radius=
m  # Print the map
```

Conclusions

The map suggests the following conclusions, though more detailed analysis is needed to confirm them.

- The majority of the stations show a warming trend
- Some stations show a cooling trend.
- Qualitatively from the map, it looks like the south/south east is warming at almost all stations.

Future questions

- Is the trend related to latitude?
- Are the fits sensitive to the exact time-range used?
- What does Alasksa look like?